TRANSFORMASI *CHANNEL SYSTEM* KE *LABELLED TRANSITION SYSTEM*

Siti Mutmainah[1] dan Reza Pulungan[2]
Jurusan Ilmu Komputer dan Elektronika
Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Gadjah Mada
FMIPA Gedung Selatan, Sekip Unit III, Kotak Pos BLS. 21
55281 Yogyakarta, Indonesia
Telp. (+62 274) 546194, Faks. (+62 274) 546194
E-mail: [1]Siti.mutmainah@mail.ugm.ac.id

Abstract

A parallel system must be developed with extra precision to achieve a high level of dependability to produce error-free software.
Therefore, we need an appropriate and formal model of the system. The model is necessary for verification software in model checking. A model that is often used in model checking is Labelled Transition System (LTS). An LTS can be generated from Program Graphs (PG), which are representations of programs in any programming language. However, in modeling a parallel system we need to describe the communications(model of interactivity) between the Program Graphs. For this purpose, we need and use Channel Systems (CS).
Therefore, we often need to transform CS to LTS. The aim of this research is to developa transformation algorithm to generate an LTS from a given CS. Through the concept of parallelism, the algorithm is constructed by exploiting the so-called Structured Operational Semantic (SOS) of the CS.
The transformation algorithm is implemented by developing a prototype tool designed with 3 essential components:input, process and output. Input is a text file that contains a model of a parallel system in the form of a CS. The model will be processed by using thetransformation algorithm that generates an LTS as output text file.
Functionally, the algorithm can handle models of parallel systems written ina CS that consists of several PGs containing both looping and branching. Moreover, it has ability to satisfy various parallelism models such as interleaving, synchronous and asynchronous.

***Keywords**: Channel System (CS), Labelled Transition System (LTS), parallel, Program Graph (PG), statesemantic.*

A. Pendahuluan

Perangkat lunak yang bebas dari kesalahan adalah perangkat lunak yang dengan tepat mengikuti spesifikasinya (Sommerville, 2001: p.22). Oleh karena itu diperlukan spesifikasi sistem yang tepat dan formal untuk mendefinisikan sistem yang akan diimplementasikan. Keberadaan semantik formal sangat berguna untuk menghasilkan deskripsi yang tidak ambigu dari sebuah model sistem reaktif. Hal ini berguna untuk mengetahui tingkat *correctness* dari sistem tersebut. Correctness dapat dicapai dengan

teknik verifikasi. Dalam verifikasi ada dua metode yang sering digunakan, yaitu *theorem proving* dan *model checking*.

Model checking merupakan teknik efektif untuk menemukan kesalahan desain yang potensial terjadi. Teknik ini menggunakan metode formal untuk memverifikasi semua state yang terhingga banyaknya dalam sebuah sistem dan memverifikasi kebenaran dari spesifikasi sistem yang diekspresikan dalam logika temporal. Model checking bekerja secara atomatis dan relatif cepat. Jika terdapat error dalam desain, model checking akan memproduksi sebuah *counterexample* yang digunakan sebagai penunjuk sumber error (Baier dan Katoen, 2008: p.12).

Hampir semua tool model checking menggunakan Labelled Transition System (LTS) dalam pemrosesannya. Hal ini dikarenakan LTS merupakan bentuk universal dan mempunyai semantik yang jelas. LTS digunakan sebagai dasar untuk mendeskripsikan behavior proses-proses yang terjadi dalam sistem. LTS dapat dihasilkan melalui program graph (PG). Program graph adalah suatu formalisme yang dapat merepresentasikan semua program yang ditulis dalam bahasa pemrograman apapun. Namun karena yang akan dimodelkan adalah sistem reaktif, maka harus ada model komunikasi untuk menggambarkan interaktifitas antar proses. Oleh karena itu dibutuhkan sebuah *channel system*(CS) sebagai media ketika terjadi proses pengiriman (*sending*) atau penerimaan (*receiving*) nilai yang melibatkan variabel-variabel yang dimiliki oleh beberapa program graph. Bentuk LTS yang dihasilkan akan sangat bermanfaat dalam pengecekan model, sehingga penulis memandang perlu adanya sebuah tool otomatis untuk menghasilkan bentuk LTS dari channel system.

Dalam penelitian Tretmans (2008: p.1-38) dideskripsikan model dan bahasa yang digunakan, seperti LTS dan beberapa variantnya untuk memodelkan spesifikasi, implementasi dan testing. Penelitian ini juga secara formal mendefinisikan relasi implementasi *ioco* yang mengekspresikan apakah implementasi sudah sesuai dengan spesifikasinya atau belum. Sedangkan De Nicola dan Loreti (2007: p.133-146) pernah melakukan penelitian di mana LTS dijadikan sebagai model fundamental untuk

membuktikan kesesuaian *property* dari sistem yang berjalan secara *concurrent*. Dalam penelitiannya mereka mengenalkan MLTS (*Multiple Labelled Transition System*) sebagai generalisasi dari LTS yang memiliki fitur yang sangat penting ketika mempertimbangkan bahasa dan model untuk pemrograman network. Dijelaskan pula mengenai bagaimana MLTS dapat digunakan untuk mendeskripsikan semantik operasional dari calculus untuk mobilitas yang dikenal dengan *asynchronous π-calculus* (Boudol, 1992: p.15). Selain itu Uchitel et al. (2003: p.19-27) pernah menulis bahwa *state machine* seperti LTS secara umum digunakan untuk mendeskripsikan behavior dari sebuah sistem. Formalisasi tersebut diasumsikan menjadi deskripsi yang lengkap dari behavior sebuah sistem pada level abstraksi (Keller, 1976: p.371–384). Penelitian yang mereka lakukan dengan menggunakan PLTS (*Partial Labelled TransitionSystem*) untuk mengambil apa saja yang belum terdefinisikan dari behavior sebuah sistem. PLTS merupakan perluasan dari LTS yang secara tidak langsung memodelkan *action* yang kemungkinan tidak pernah terjadi dalam masing-masing state. Mereka menggunakan pendekatan Whittle dan Schumann (Whittle dan Schumann, 2000: p.314-323).

Dalam penelitian ini penulis bermaksud untuk memproduksi LTS dengan cara yang berbeda yakni dengan mentransformasi CS. CS digunakan untuk menggambarkan protokol komunikasi di mana channel itu sendiri memiliki buffer *first-in, first-out* yang berisi nilai atau *message*. Proses yang berkomunikasi dengan proses lain di dalam sistem dapat melalui channel tersebut.

B. Landasan Teori

Labelled Transition *System* (LTS)

Labelled Transition System (LTS) adalah model dasar untuk merepresentasikan reaktifitas, kongkurensi dan komunikasi sistem dan sering digunakan dalam ilmu komputer sebagai model untuk menggambarkan behavior dari sistem. Definisi menurut Baier dan Katoen (2008) bahwa sebuah Labelled Transition System adalah sebuah tuple (S, Act, →, I, AP, L), di mana:

- S adalah himpunan dari *state*.
- *Act* adalah himpunan dari *action*.
- S × Act × S yang merupakan sebuah relasi transisi.
- I adalah himpunan bagian S (I S) dan merupakan himpunan dari *state* inisial.
- AP adalah himpunan dari *atomic proposition*.
- L : S 2^{AP} adalah sebuah fungsi *labeling*.

Program *Graph*

Program Graph (PG) adalah bentuk representasi dari program yang menjalankan sebuah proses. Proses tersebut digambarkan dalam bentuk graph yang terdiri atas lokasi awal, lokasi akhir dan transisi yang menunjukkan adanya perpindahan dari lokasi awal ke lokasi akhir. Definisi menurut Baier dan Katoen (2008) bahwa secara formal definisi sebuah Program Graph adalah sebuah tuple (Loc, Act, Effect, →, Loc0, g0), di mana:

- Loc adalah set dari lokasi yang terbatas jumlahnya.
- Act adalah set dari action.
- Effect: Act × Eval(var) → Eval(var); misalnya jika α adalah x := x + y maka Effect(α, [x = 1, y = 7]) = [x = 8, y = 7]. Jadi, Effect di sini menggambarkan sebuah akibat yang terjadi terhadap perubahan nilai variabel jika action α dijalankan.
- → adalah himpunan bagian dari Loc × Cond(Var) × Act × Loc, yaitu relasi transisi kondisional.
- Loc0 adalah himpunan bagian dari Loc yang merupakan set dari inisial lokasi.
- g0 adalah elemen dari Cond(Var) yang merupakan inisial dari kondisi.

Channel System

Dalam sistem, sebuah proses akan berkomunikasi dengan proses lain dengan menggunakan channel. Menurut Baier dan Katoen (2008) channel system adalah sebuah cara untuk menggambarkan protokol komunikasi yang terjadi dalam sebuah sistem yang dapat ditulis dengan: